UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/087,376	03/01/2002	John B. Duffie III	112025-0488	3382

24267 7590 03/26/2008
CESARI AND MCKENNA, LLP
88 BLACK FALCON AVENUE
BOSTON, MA 02210

EXAMINER	
DAFTUAR, SAKET K	

ART UNIT	PAPER NUMBER
2151	

MAIL DATE	DELIVERY MODE
03/26/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/087,376
Filing Date: March 01, 2002
Appellant(s): DUFFIE ET AL.

James A. Blanchette, Registration Number 51,477
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 02/05/2008 appealing from the Office action mailed 09/05/2007.

(1) Real Party in Interest

The real party in interest in this brief is Cisco Technology Inc.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

6370560	Robertazzi et al.	04-2002
6,587,866	Modi et al	07-2003
6,065,046	Feinberg et al.	05-2000

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

Claims 1-9, 13-14, 17, 21-27, 31 and 32 are rejected under 35 U.S.C. 102(e) as being anticipated by Robertazzi et al. U.S. Patent Number 6,370,560 B1 (hereinafter Robertazzi).

As per claim 1, Robertazzi discloses determining the size of the packet (see column 4, line 7 – 59); a cost associated with the packet, the cost representing a load associated with processing the packet (see column 4, line 7 – 59; column 11, lines 5 – 28); determining an anticipated load for each coprocessor in the plurality of coprocessors using the cost (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39); and selecting the coprocessor [selected processors] from the plurality of coprocessors based on the anticipated load (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39).

As per claim 2, Robertazzi discloses calculating the cost using a rate associated with processing the packet (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39).

As per claim 3, Robertazzi discloses the rate is stored in a lookup table (see Figure 9, column 19, lines 1-19).

As per claims 4 and 5, Robertazzi discloses dividing [divided load] the packet's size by the rate (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39). Robertazzi also discloses the step of multiplying the packet's size [multiplied by the size of overall load] by a multiplicative inverse of the rate (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39).

As per claim 6, Robertazzi discloses applying the packet's size to a lookup table containing one or more cost values to determine the cost (see Figure 9, column 19, lines 1-19).

As per claim 7, Robertazzi discloses adding the cost to a cumulative load associated with each coprocessor in the plurality of coprocessors (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39).

As per claim 8, Robertazzi discloses selecting the coprocessor from a group of one or more coprocessors whose anticipated load is a minimum load [minimum or low resource utilization cost] (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39).

As per claim 9, Robertazzi discloses the coprocessor is selected using a scheduling [optimization techniques can be applied to generic problems scheduling, graph problems, sequencing, and assignment] algorithm (column 20, lines 10-19).

As per claims 13 and 14, Robertazzi discloses of incrementing a cumulative load associated with the selected coprocessor (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39).

Robertazzi also discloses adding the cost to the cumulative load (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39).

As per claim 17, Robertazzi discloses a memory containing one or more software routines [optimization routine, column 7, lines 9-18; column 11, line 5 – column 12, line 39, memory to store operations], including a software routine configured to determine the size of the packet (see column 4, line 7 – 59), a cost associated with the packet of that size (see column 4, line 7 – 59; column 11, lines 5 – 28), the cost representing a load associated with processing the packet (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39); and a processor configured to execute the software routines [optimization routine, column 7, lines 9-18; column 11, line 5 – column 12, line 39, memory to store operations] to determine an anticipated load for each coprocessor in the plurality of coprocessors using the cost and to select the coprocessor [selected processors] from the plurality of coprocessors based on the anticipated load (see column 4, line 7 – 59, column; column 11, line 5 – column 12, line 39).

As per claim 21, claim 21 is device claim of method claim 1. They do not teach or further define over the limitation as recited in claim 1. Therefore, claim 21 rejected under same scopes as discussed in claim 1, supra.

As per claim 22, claim 22 is computer readable media claim of method claim 1. They do not teach or further define over the limitation as recited in claim 1. Therefore, claim 22 rejected under same scopes as discussed in claim 1, supra.

As per claims 23- 25, claims 23 – 25 are method claims of claims 1, 2-3 and 6. They do not teach or further define over the limitation as recited in claims 1, 2-3 and 6. Therefore, claims 23 – 25 are rejected under same scope as recited in claims 1, 2-3 and 6, supra.

As per claim 26, Robertazzi discloses determining a cumulative load for each coprocessor (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39), the cumulative load representing load due to packets currently awaiting (see column 5, line 47 – column 6, line 36, job queue determining resource utilization cost of any available distributed processor connected to the system's network and provides the controller with divisible load or task to be distributed) processing at that coprocessor; determining a size of the received packet (see column 4, line 7 – 59); determining a cost for processing the received packet at each coprocessor, the cost determined, at least in part, in response to the size of the received packet and a processing rate of that coprocessor (see column 4, line 7 – 59; column 11, lines 5 – 28); combining the cumulative load and the cost at each coprocessor, to create an anticipated load for each coprocessor (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39); comparing the anticipated loads [compare different processors using the same parameters]of

all the coprocessors (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39); and selecting, in response to the comparing, a particular coprocessor [selected processors] of the plurality of coprocessors to perform the processing operation on the received packet (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39).

As per claim 27, they do not teach or further define over the limitation as recited in claim 8, respectively. Therefore, claim 27 rejected under same scope as recited in claims 8, supra.

As per claim 31, Robertazzi discloses A plurality of queues configured to store packets currently awaiting processing, each queue associated with one of the coprocessors, each queue associated with a cumulative load that represent a load to process packets in that queue (see column 5, line 47 – column 6, line 36, job queue determining resource utilization cost of any available distributed processor connected to the system's network and provides the controller with divisible load or task to be distributed);a data structure configured to store processing rates, each processing rate associated with one of the coprocessors; and a processor configured to determine a size of the received packet, and in response to the size of the received packet (see column 4, line 7 – 59), and the processing rate of each coprocessor, determine a cost to perform a processing operation on the received packet at each coprocessor (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39), the processor further configured to combine the cost at each coprocessor with the cumulative load at that

coprocessor to create an anticipated load at each coprocessor (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39), and to select a particular coprocessor to perform the processing operation on the received packet in response to comparison of the anticipated load at each coprocessor (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39).

As per claims 32, they do not teach or further define over the limitation as recited in claims 8, 10-12, and 13-16, respectively. Therefore, claims 32, 33, and 34 are rejected under same scope as recited in claims 8, 10-12, and 13-16, supra.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 10-12, 15-16, 18-20, 28-29, 30,33,34, and 35 rejected under 35 U.S.C. 103(a) as being unpatentable over Robertazzi et al, U.S. Patent Number 6,370,560 B1 (hereinafter Robertazzi) in view of Modi et al, U.S. Patent Number 6,587,866 B1 (hereinafter Modi).

Robertazzi discloses a plurality of queues configured to store packets currently awaiting processing, each queue associated with one of the coprocessors, each queue associated with a cumulative load that represent a load to process packets in that queue; a data structure configured to store

processing rates, each processing rate associated with one of the coprocessors; and a processor configured to determine a size of the received packet, and in response to the size of the received packet, and the processing rate of each coprocessor, determine a cost to perform a processing operation on the received packet at each coprocessor, the processor further configured to combine the cost at each coprocessor with the cumulative load at that coprocessor to create an anticipated load at each coprocessor, and to select a particular coprocessor to perform the processing operation on the received packet in response to comparison of the anticipated load at each coprocessor.

However, Robertazzi is silent about determining if a port associated with the packet is congested, and selecting coprocessor based on anticipated load not a minimum load.

As per claim 10, Modi teaches that determining if a port associated with the packet is congested (see column 10, lines 56-58).

Therefore, it would have bee obvious to one having ordinary skilled in the art at the time the invention was made to combine teaching of Robertazzi and Modi as they both are from same field endeavor to provides a method of distributing packets to server nodes in cluster of nodes including the steps of receiving a packet that is directed to a selected service which can be provided by plurality of nodes in cluster and determining an appropriate server node based at least in part on whether the service designates scalable service between the

server nodes that is efficient, scalable and highly available, and allows client affinity.

As per claim 11, Robertazzi discloses that selecting the coprocessor from a group of one or more coprocessors whose anticipated load is not a minimum load (see column 4, line 7 – 59; column 11, line 5 – column 12, line 39).

As per claim 12, claim 12 falls under the same limitation of claim 8. Therefore, claim 12 has been rejected under same rationale.

As per claims 15 and 16, Modi teaches decrementing a cumulative load associated with the selected coprocessor (see column 12 lines 54-59). Additionally, Modi also discloses subtracting the cost from the cumulative load [examiner consider deletion of connection and deleting service on particular nodes as removing service weight from that node].

Therefore, it would have bee obvious to one having ordinary skilled in the art at the time the invention was made to combine teaching of Robertazzi and Modi as they both are from same field endeavor to provides a method of distributing packets to server nodes in cluster of nodes including the steps of receiving a packet that is directed to a selected service which can be provided by plurality of nodes in cluster and determining an appropriate server node based at least in part on whether the service designates scalable service between the server nodes that is efficient, scalable and highly available, and allows client affinity.

As per claim 18, Modi teaches a data structure (see column 4, lines 41); wherein the cost is determined using information contained in the data structure (see column 4, lines 41; see column 7, lines 40-44).

Therefore, it would have bee obvious to one having ordinary skilled in the art at the time the invention was made to combine teaching of Robertazzi and Modi as they both are from same field endeavor to provides a method of distributing packets to server nodes in cluster of nodes including the steps of receiving a packet that is directed to a selected service which can be provided by plurality of nodes in cluster and determining an appropriate server node based at least in part on whether the service designates scalable service between the server nodes that is efficient, scalable and highly available, and allows client affinity.

As per claim 19, Modi teaches that the information contained in the data structure includes the cost (see column 4, lines 41; see column 7, lines 51-54).

As per claim 20, Modi teaches that the information contained in the data structure includes a rate the coprocessor can process the packet (see column 4, lines 41; see column 7, lines 19-20).

As per claims 28, 29, 33, and 34 they do not teach or further define over the limitation as recited in claims 10-12, and 13-16, respectively. Therefore, claims 27, 28 and 29 are rejected under same scope as recited in claims 10-12, and 13-16, supra.

a. Claims 30 and 35 are rejected under 35 U.S.C. 103(a) as being unpatentable over Robertazzi and Modi as applied to claims 1-29 and 31-34 above, and further in view of Feinberg et al, U.S. Patent No. 6,065,046 (hereinafter Feinberg).

Robertazzi in view of Modi discloses determining a cumulative load for each coprocessor, the cumulative load representing load due to packets currently awaiting processing at that coprocessor; determining a size of the received packet; determining a cost for processing the received packet at each coprocessor, the cost determined, at least in part, in response to the size of the received packet and a processing rate of that coprocessor; combining the cumulative load and the cost at each coprocessor, to create an anticipated load for each coprocessor; comparing the anticipated loads of all the coprocessors; and selecting, in response to the comparing, a particular coprocessor of the plurality of coprocessors to perform the processing operation on the received packet.

However, neither Robertazzi nor Modi discloses an encryption operation for processing packets.

Feinberg teaches the processing operation is an encryption operation (see column 5,lines 21-67).

Therefore, it would have bee obvious to one having ordinary skilled in the art at the time the invention was made to combine teaching of Robertazzi, Modi and Feinberg as they all are from same field endeavor to provides a secure

encrypted method of distributing packets to server nodes in cluster of nodes including the steps of receiving a packet that is directed to a selected service which can be provided by plurality of nodes in cluster and determining an appropriate server node based at least in part on whether the service designates scalable service between the server nodes that is efficient, secure, scalable and highly available, and allows client affinity.

(10) Response to Argument

Appellant's arguments filed 02/05/2008 have been fully considered but they are not persuasive. Appellant continues to argue in substance that:

i. First, Robertazzi does not teach the Applicant's claimed "determining a cost associated with the packet ... the cost representing a load associated with processing the packet."

ii. Second, Robertazzi does not teach the Applicant's claimed "determining an anticipated load for each coprocessor" and "selecting the coprocessor from the plurality of coprocessors based on the anticipated load."

iii. Robertazzi does not teach determining an anticipated load by "adding the cost to a cumulative load associated with each coprocessor in the plurality of coprocessors."

In response to appellant above arguments i) – iii), examiner considers Robertazzi discloses "a load sharing system which minimizes overall costs by assigning segments of a divisible load to distributed processor platforms based

on the resource utilization cost of each processor platform. The distributed processor platforms are connected via data links which also have associated resource utilization costs. A controller divides a divisible load or task and assigns each segment of the load or task to a processor platform **based on the processor platform's resource utilization cos**t and data link cost. After the initial allocation, an optimizing reallocation is performed to reduce the overall monetary cost processing the load or task. "

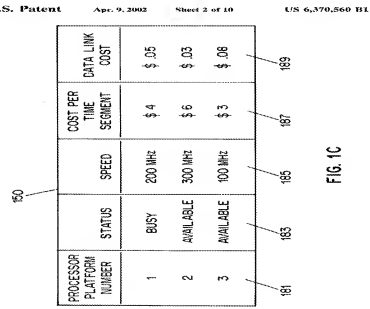

FIG. 1C

Robertazzi clearly discloses determining costs associated with packet at different transmission speeds are different and determining a load associated with processor. Robertazzi discloses "a divisible load or task can be segmented and have its portions be simultaneously processed on different types of computer

platforms including a 486 computer, a work station and a supercomputer. Each of the individual processors or group of processors could be a computer "utility" which leases computer time to other users. [Loads and tasks distributed on plurality of processor].

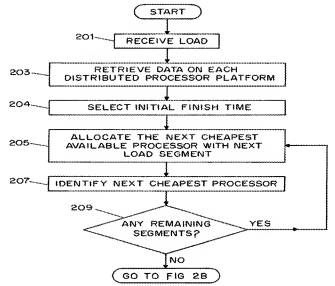

FIG. 2A

Each different type of processor platform has an associated **resource utilization cost** when used to process a portion of a load **[Each processor has a different utilization cost, please note, utilization cost, not monetary cost].** The platform includes the processor, supporting electronics and devices necessary for the processor's operation (e.g., the "motherboard"). **The resource utilization cost of a processor includes factors such as the monetary computing cost**

whose unit is "cost per second" and the computing speed whose unit is "loads/tasks per second".

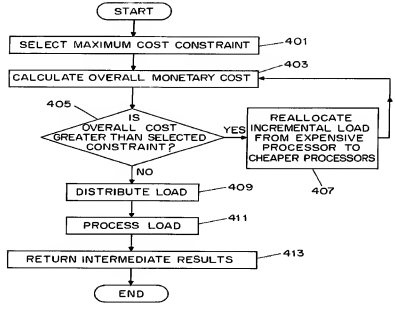

FIG. 4

The unit "load/task" in the computing speed parameter is of predetermined size or task magnitude in order to be able to compare different processors using the same parameters...**The resource utilization cost of a processor is based upon a number of factors which could include the operating cost of the**

processor platform, the initial purchase price of the processor platform, and the data link costs between the distribution processor and the controller processor." (Column 4, lines 7-59).

Robertazzi clearly discloses selecting a coprocessor based on anticipated load. In column 11,line 5 - column 12, line 39, Robertazzi discloses Figure 4 of the invention where Robertazzi discloses "calculating overall cost of the divisible job to be distributed among the available processor platform in the network." Robertazzi also discloses that "**Task size can be computed based on such factors as the complexity of the task and the necessary data inputs.** The cost is minimized by placing as much of **the load or task on the processor platforms** with associated data links having a **relatively low resource utilization cost** while still completing the processing within an acceptable finish time" and compares "**the calculated overall cost is greater than the selected cost constraint" to make determination to "**reallocates **an incremental portion of the load from the most expensive processor platform to the cheaper processor platforms (which reduces cost but most likely decreases speed) and also "**distributes the load to the selected processor platforms using the determined allocation in the previous steps."

Therefore, it is clear that Robertazzi discloses "determining a cost associated with the packet ... the cost representing a load associated with processing the packet and determining an anticipated load for each coprocessor" and "selecting the coprocessor from the plurality of coprocessors based on the anticipated load

and adding the cost to a cumulative load associated with each coprocessor in the plurality of coprocessors."

Appellant continues to argue in substance that:

iv. Neither Robertazzi nor Modi teach the Applicant's claimed "decrementing a cumulative load associated with the selected coprocessor."

v. Neither Robertazzi nor Modi teach the Applicant's claimed "determining if congestion is present at an output port associated with the received packet, and if congestion is present, selecting a coprocessor with non-minimum anticipated load to perform the processing operation on the received packet."

vi. Neither Robertazzi nor Modi teach the Applicant's claimed "determining, for each coprocessor, sizes of the packets currently awaiting processing at that coprocessor and using the sizes in conjunction with the processing rate of that coprocessor to determine the cumulative load."

In response to appellant argument iv) – vi), Robertazzi discloses a plurality of queues configured to store packets currently awaiting processing, each queue associated with one of the coprocessors, each queue associated with a cumulative load that represent a load to process packets in that queue; a data structure configured to store processing rates, each processing rate associated

with one of the coprocessors; and a processor configured to determine a size of the received packet, and in response to the size of the received packet, and the processing rate of each coprocessor, determine a cost to perform a processing operation on the received packet at each coprocessor, the processor further configured to combine the cost at each coprocessor with the cumulative load at that coprocessor to create an anticipated load at each coprocessor, and to select a particular coprocessor to perform the processing operation on the received packet in response to comparison of the anticipated load at each coprocessor.

Modi on the other hand teaches "a scalable cluster system that provides scalable services for client applications is provided with client affinity. The scalable services are transparent to the client application. To facilitate this transparent scalability, the system provides different types of client affinity to different services for the client applications. Services may have no client affinity, so that different packets sent during different connections could be sent to different nodes on the cluster. Services may have single service client affinity, which causes packets for a single service from different connections from the same source to be sent to the same node. Services may have multiple service client affinity, which causes packets for different services from different connections from the same source to be sent to the same node. Services may have wild card client affinity, which causes packets for different destination ports from different connections from the same source to be sent to the same node." (Abstract).

Modi teaches a load balancing policy is included while maintaining a client affinity when processing packets from single and multiple service(s) from different connection to be sent to the same node and port number.

"If the service is a scalable service, the system determines to which server node to send the packet. In doing so, the system first determines whether the service group associated to the service of the packet has a load balancing policy type with client affinity (step 605), i.e., whether the load balancing policy type is client affinity or wild card client affinity. If so, the system hashes the client IP address over PDT 304 to select a bucket from PDT 304 (step 606).

If not, the system hashes the client IP address and the port number to select a bucket from the PDT 304 (step 607). It should be noted that when the policy type is a client affinity policy, only the client address is hashed (as opposed to both the client address and port number). This is important in any systems where a single source may have multiple parallel connections with a server that needs to combine the information from the parallel connections **(as for example while listening to an internet broadcast, one connection may be used to receive the broadcast and another connection may be used to control the broadcast.)** When client affinity is not required, hashing both the client address and client port statistically tends to provide better load balancing.

Next, the system determines whether the protocol is TCP (step 608). If the protocol is not TCP (meaning it is UDP), the system retrieves an identifier for a server node from the entry in the selected bucket of the PDT(step 612).

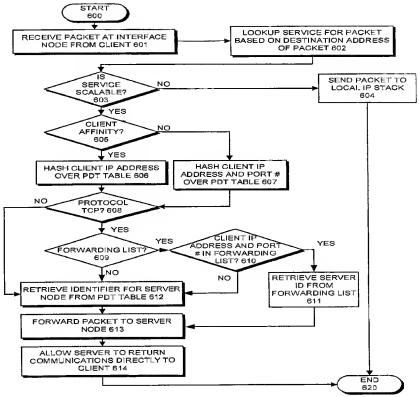

FIG. 6

Otherwise, if the protocol is TCP, the system determines whether the selected bucket of the PDT 304 has a forwarding list (step 609). If the selected bucket does not have a forwarding list, the system retrieves the identifier for the server node from the entry in the selected bucket of the PDT 304 (step 612). If the selected bucket has a forwarding list, the system determines if the client IP address and port number are in a forwarding list (step 610). The forwarding list

allows existing TCP connections to continue to go to the same node, when the PDT 304 is changed. If so, the system retrieves the server identifier from the forwarding list (step 611). Otherwise, the system retrieves the server identifier from the selected bucket in PDT 304 (step 612). In the preferred embodiment, where the forwarding list and a copy of the PDT is maintained in a service object, only the client IP address and client port of a listing in the forwarding list need to be compared with the client IP address and the client port of the packet to determine if there is a match, since the matching of the packet with the service object has already matched the service IP address and service port. "

Therefore, one having ordinary skill in the art would know the importance of having a single source that includes a multiple parallel connection with server that needs to combine the information from the parallel connection to avoid any traffic related congestion and collision associated with that particular single source or port. Therefore, Robertazzi in view of Modi clearly discloses determining if congestion is present at an output port associated with the received packet, and if congestion is present, selecting a coprocessor with non-minimum anticipated load to perform the processing operation on the received packet. Therefore, it would have been obvious to one having ordinary skilled in the art at the time the invention was made to combine teaching of Robertazzi and Modi as they both are from same field endeavor to obtain a predictable result to provides distributing packets to server nodes in cluster of nodes including the steps of receiving a packet that is directed to a selected service which can be

provided by plurality of nodes in cluster and determining an appropriate server node based at least in part on whether the service designates scalable service between the server nodes that is efficient, secure, scalable and highly available, and allows client affinity.

In addition, Modi also discloses about maintaining a forwarding list while maintaining the load balancing policy in figure 10 "FIG. 10 illustrates a method of generating or updating a forwarding list when a server identifier of a bucket in a PDT is changed. Such changes may be made for various reasons. One reason for changing a server identifier is to change the load balancing for a PDT of a service group. An operator may change the load balancing between nodes as a way of tuning the system.

When a server identifier of a bucket of a PDT is changed, a server identifier for an old node is replaced with a server identifier of a new node (step 1004). The interface node 103 checks to see if the old node has any existing TCP connections (steps 1008 and 1010). If there are no existing TCP connections, the process is done (step 1022). If the old node has existing connections, a query is made to see if the bucket has a forwarding list (step 1012). If the bucket does not have a forwarding list, a forwarding list is created (step 1014). Both branches of step 1012 then add the existing TCP connections to the forwarding list (step 1016). In one embodiment, all existing TCP connections for packets with the same service IP address and service port as the PDT are added to the forwarding list. In another embodiment, only those TCP

connections with the same service IP address and service port as the PDT and combination source IP address and source port that can be hashed into the bucket are added to the forwarding list, so that only connections corresponding to the bucket are added to the forwarding list.

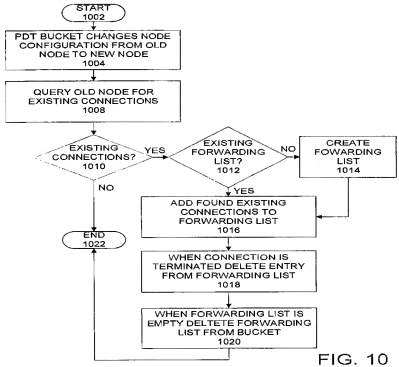

FIG. 10

The advantage of placing all connections in the forwarding list is that it saves time required by hashing to see which source IP address and source port

corresponds to the bucket. The advantage of only adding connections corresponding to the bucket to the forwarding list is that it keeps the forwarding list size to a minimum.

When an existing TCP connection is terminated, the old node sends a message indicating that the connection has been terminated, and the connection is deleted from the forwarding list (step 1018). When the forwarding list becomes empty the entry for the forwarding list may be removed from the bucket (step 1020). "

Therefore, one having ordinary skill in the art would know that it would have been obvious that adding connection in network load balancing would be an adding load in network. Therefore, Modi in view of Robertazzi discloses decrementing a cumulative load associated with the selected coprocessor, selecting a coprocessor with non-minimum anticipated load to perform the processing operation on the received packet and *determining, for each coprocessor, sizes of the packets currently awaiting processing at that coprocessor and using the sizes in conjunction with the processing rate of that coprocessor to determine the cumulative load.* Therefore, it would have been obvious to one having ordinary skilled in the art at the time the invention was made to combine teaching of Robertazzi and Modi as they both are from same field endeavor to obtain a predictable result to provides a method of distributing packets to server nodes in cluster of nodes including the steps of receiving a packet that is directed to a selected service which can be provided by plurality of

nodes in cluster and determining an appropriate server node based at least in part on whether the service designates scalable service between the server nodes that is efficient, scalable and highly available, and allows client affinity.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/Saket K Daftuar/

Examiner, Art Unit 2151

Conferees:

/John Follansbee/

Supervisory Patent Examiner, Art Unit 2151

/Jason D Cardone/
Supervisory Patent Examiner, Art Unit 2145